A Survey of Parallel Programing Tools

Doreen Y. Cheng[1]

Report RND-91-005, May 1991

RND Branch
NAS Systems Division
NASA Ames Research Center
Mail Stop 258-6
Moffett Field, CA 94035-1000

---

[1] Computer Sciences Corporation, NASA Contract NAS 2-12961, Moffett Field, CA 94035

Introduction

This survey examines thirty-nine parallel programming tools. Focus is placed on those tool capabilities needed for parallel scientific programming rather than for general computer science. The tools are classified with current and future needs of NAS in mind: in particular, existing and anticipated NAS supercomputers and workstations, operating systems, programming languages and applications. They are divided into four categories: suggested acquisitions, tools already brought in, tools worth tracking, and tools eliminated from further consideration at this time.

Section 1 lists the tools that are suggested acquisitions. They have been ranked according to the functions provided, maturity at the time of this survey, and suitability of building new functions on top of them. Some of them have overlapping functions. They have been listed in order, most preferred first; price was not considered in this ranking.

No ordering has been placed on the listing for the remaining sections. Section 2 presents the tools that have already been brought to NAS, but which are not yet available to the users. They were obtained either because they were in the public domain, or some of their functionality was desired by NAS parallel tool developers. Of these tools, only Schedule, Force, ParaScope, Axe/Aims, Parti, Hypertask, and CRAY/fpp are of current interest to NAS.

Section 3 lists the tools that are of interest to NAS, but not suggested acquisitions at this time. Some of them are in an early, but active, research stage. Others have been discarded after being evaluated at NAS; they may become very attractive if new development directions are taken. It would be prudent to keep track of their development.

The tools in Section 4 are eliminated from further consideration at the present time. Some of these tools are eliminated because they are not suitable for NAS applications or NAS platforms. Others are eliminated because it is difficult to contact and collaborate with their developers.

The functions and maturity of each tool are evolving as well as the interests of NAS users and parallel tool developers. Except for Axe/Aims, the information collected in this survey reflects the status up until February, 1991. Changes in Axe/Aims between February, 1991 and May, 1991 have been included.

To make it easier for readers to find a specific tool in the report, an alphabetized look-up table is included at the beginning. The table indicates the usefulness of a tool to NAS as well as the page and section where the tool is described.

Four tables are appended at the end of the report. These tables compare the features and status of the tools that are of immediate interest to NAS (i.e. those in Sections 1 and 2 and some in Section 3). The first table lists the functions of the tools that transform a sequential program into a parallel one. The second table covers the functions that help users write parallel programs. The third tables lists the functions that assist in parallel debugging and performance optimization. The last table lists the availability and system requirements of the tools.

In this report, OS means operating system and GUI represents graphic user interface.

Index

1. Suggested Acquisitions

1.1 Forge/MIMDizer

Function of Forge:

- Program instrumentation
- Dependency analysis
- DO-loops parallelization guided by run time profiling
- Parallel/vector directives insertion
- Queries about the program (use-def, call-tree, common use, routine interface consistency, timing variable trace, etc)
- Code restructuring (invert loop, split loop, merge loop, collapse nested loops, unroll loop, inline routine, remove secondary induction variables, etc)
- A database on which new tools can be developed

Function of MIMDizer:

- All functions of Forge available
- Language extension to Fortran for programming message-passing machines
- Code frames generated for specification of communication and synchronization between code blocks written in Fortran (specification automatically transformed into message passing)
- Consistency checking for message-passing, variable definition, and caller-callee argument passing
- Automatic decomposition of arrays and static distribution of loops onto processors
- Automatic and dynamic repartitioning of arrays when needed
- Monitoring facilities for performance tuning and debugging

Evaluation of Forge by Doreen Cheng through use:

- The interactive nature provides convenient access to the tools and to information about the program.
- Preliminary evaluation showed the user response time is faster than that of other tools with similar functionality.
- The database is designed for program analysis. The size of the Forge database is only 3 to 4 times the size of the source code. Its speed is noticeably faster.
- Many bugs were encountered but PSR fixed them quickly.
- Forge flags a converted loop as parallel by simply using a DO ALL compiler directive. This has resulted in significant performance degradation which cancels the potential speedup by parallelization.
- Forge lacks the capabilities to guide dependence elimination, code transformation and parallelization.
- Vectorization and parallelization functions do not use a common user interface and do not allow mixed usage.

Evaluation of MIMDizer by Doreen Cheng through literature search and phone contacts with the vendor:

- Extensive set of tools for programming message-passing machines.
- Alpha release.
- Lacks automatic load balancing.

Platform for Forge: Y-MP, X-MP, CRAY2, Sun, IRIS, NEC (negotiating)

Platform for MIMDizer: iPSC, Sun, IRIS

OS: UNIX

Language: Fortran

GUI: X-Windows, Sunview

Cost*: $28,050/workstation; $4,500/yr maintenance
Facility license**: $93,500
Unlimited license***: $158,950, $18,000/yr maintenance

Supplier: Pacific Sierra Research

Contact: John Levesque, (916) 621-1600

* Numbers quoted are for NASA Ames Research Center.

** One of two options available:

(1) Software available for running on up to 10 workstations.
(2) X-Window and/or terminal version for running on a single-host communicating with as many workstations (running X-Windows) or terminals emulating VT100.

*** Unlimited workstations and unlimited hosts running X-Windows and/or terminal version.

See reference [1] [2] [3]

1.2 Express

Function:

- Library calls provided for parallelization
- Program instrumentation
- Hardware configuration management
- Automatic loop parallelization
- Automatic data distribution and domain decomposition
- Run time profile used for guidance
- Dynamic load balancing
- Parallel I/O
- Interactive memory access visualization
- Post-mortem communication and event analysis
- Communication and event monitoring
- Parallel algorithm monitoring
- Source level debugger
- Deadlock detection
- Converts Fortran 90 source code to Fortran 77
- A database used for run time information

Evaluation by Doreen Cheng through literature search and phone contacts with the vendor:

- Provides extensive set of tools for message-passing machines (debugging, performance monitoring, load balancing and parallelization).
- Covers a broad range of hardware, operating system, and languages.
- Debugger, profiler, parallel I/O, graphics are not available on Y-MP, and may not be in the future.
- Lacks support for interactive dependency analysis.

Platform: iPSC, Y-MP (beta), NCUBE, Sun, PC, Macintosh

OS: UNIX, DOS, Macintosh

Language: Fortran 77, Fortran 90, C, C++

GUI: X-Windows, Sunview, Postscript

Cost: $3,000 per Intel iPSC/860
$15,000 per Y-MP
$1,500 for network of Suns
20% maintenance fee per year

Supplier: ParaSoft

Contact: Adam Kolawa, (818) 792-9941

See reference [4]

1.3 E/SP:

Function:

- Program instrumentation
- Dependency analysis
- DO-loop parallelization
- Rough estimate of performance
- Parallel directive insertion
- Graphic editor for constructing new parallel programs
- A database for developing new tools

Evaluation by Doreen Cheng through literature search, observing demo, and phone contacts with the vendor

- The interactive nature provides a convenient access to tools and information about the program.
- For 1000 lines of code, 1 Mbytes is required for the database generated.
- It lacks support for vectorization.
- It lacks the capability to guide dependence elimination, code transformation and parallelization.

Platform:

- Sun
- In the process of negotiating with Kuck Associates for CRAY code generation
- Port to IRIS possible if desired

Language: Fortran

OS: UNIX

GUI: Sunview, X-Windows

Cost: TBD

Supplier: Scientific and Engineering Software Inc.

Contact: James. C. Browne, (512) 474-4526

See reference [5]

1.4 VecPar_77

Function:

- Dependency analysis
- Code transformation for vectorization (merge loops and if blocks, eliminate dependence, unroll loop, permute loops, etc)

Evaluation by Doreen Cheng through literature search and phone contacts with the vendor and users:

- Support for Fortran 90 is being discussed.
- Has a command-line oriented textual user interface.
- It lacks the capability to guide dependence elimination, code transformation and parallelization.
- Vectorization and parallelization tools are two separate tools.
- For large subroutines, large amounts of memory are required.

Platform: Sun, Y-MP

OS: UNIX

Language: Fortran

GUI: None

Cost: $2,500/workstation; $19,500/Y-MP
Unlimited license (use and machine) $33,510/yr
Perpetual: $78,250

Supplier: Numerical Algorithms Group Inc.

Contact: Sheila Caswell, (708) 971-2337

See reference [6]

1.5 FATCAT

Function:

- Dependency analysis

Evaluation by Doreen Cheng through literature search and phone contacts with the vendor and users:

- Able to analyze dependencies in recursive calls.
- Ported onto Y-MP in much shorter time than university products.
- Difficult to interpret the output (too much data, little guidance).
- The company is looking for funding for further development.
- For large subroutines, large amounts of memory are required.

Platform: Sun, CRAY2, X-MP, Y-MP

OS: UNIX

Language: Fortran

GUI: None

Cost: $50,000 for source license
Executable not for sale

Supplier: New Jersey Advanced Technology Inc.

Contact: David Klappholz, (201) 420-5509

See reference [7]

1.6 Faust:

Function:

- Dependency analysis
- Interactive variable tracing
- Simulator to discover the parallelism in a program
- Performance monitoring
- Interactive graphic user interface

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Tools could be integrated into NAS environment.

Platform: CRAY

OS: UNIX

Language: Fortran

GUI: X-Windows

Cost: $100 for source

Supplier: CSRD of Univ. of Illinois

Contact: David Hammerslag, (217) 244-0277

See reference [8]

1.7 IPS-2

Function:

- Performance monitoring, analysis, and visualization
- Critical path analysis and visualization

Evaluation by Doreen Cheng through literature search and e-mail with the supplier:

- Second implementation reduces program intrusiveness and storage requirements, and adds a graphic user interface.
- Tools could be integrated into NAS environment.

Platform: Y-MP (will be ready by summer 1991), Sequent Symmetry

OS: UNIX

Language: Fortran, C

GUI: X-Windows (X11)

Cost: $300 for source

Supplier: Univ. of Wisconsin

Contact: Barton P. Miller, (608) 263-3378

See reference [9]

1.8 Strand88

Function:

- Parallel language that calls code blocks written in Fortran or C
- Tools that monitor processor and communication load, and visualize the data

Evaluation by Doreen Cheng through literature search and attending seminar:

- The prolog-like language may be difficult for scientists to accept.

Platform: iPSC, Sun, Y-MP (available in spring 1991), Sequent, Encore

OS: UNIX

Language: C, Fortran

GUI: X-Windows

Cost: $22,000 for iPSC/860; maintenance $3,375/yr
$3,000/Sun; maintenance $900/yr
Site license range: $30,000 to $40,000
NAS can get 30% government discount

Supplier: Strand Software Technology Inc.

Contact: Timothy G. Mattson, (503) 690-9830

See reference [10] [11]

1.9 C-Linda

Function:

- Language extensions for parallel programming
- Source level debugger
- Program execution monitoring and visualization
- Consistency checking for tuple space usage
- Monitors message traffic and moves Linda run time library to reduce the traffic.
- Tuple space usage visualization

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Supported on a broad range of hardware platforms.
- Fortran routines can be called.

Platform: iPSC/860 (will be available shortly),
Y-MP (not fully debugged), Sun, IRIS, IBM RS6000
Apollo, Encore, Sequent

OS: UNIX

Language: C, Fortran (Fortran is not directly supported, can be called from C.)

GUI: X-Windows (for debugger)

Cost: $7,000/10 workstation
$20,000 for iPSC/2
Site license for ARC: $90,000

Supplier: Scientific Computing Associates Inc.

Contact: Ellen Smith, (203) 777-7442

See reference [12] [13]

1.10 MONMACS

Function:

- Language extensions for parallel programming
- Post-mortem performance analysis

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Has non-standard extensions.

Platform: iPSC/860

OS: UNIX

Language: C, Fortran (available in March, 91)

GUI: X-11

Cost: Public domain

Supplier: Argonne National Lab

Contact: Ewing Lusk (708) 972-7852

See reference [14]

1.11 Hypertool

Function:

- Automatic partitioning of a program for message-passing machines
- Inserts communication and synchronization primitives
- Automatic scheduling and load balancing

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Requires the user to call all routines that will be executed concurrently from the main routine.
- Retargetable compilers are planned.
- Is a research product, still buggy.

Platform: iPSC/2

OS: UNIX

Language: C

Cost: University distribution

Supplier: U.C. Irvine

Contact: Daniel D. Gajski (714) 856-4155

See reference [15]

2. Tools Already in Ames Research Center:

2.1 Schedule

Function:

- Language extension to express dependencies between code blocks written in Fortran
- Performance monitoring
- Memory access visualization
- Program execution visualization
- Critical path determination
- Program profiling
- Task scheduling
- Dynamic load balancing

Evaluation by Doreen Cheng through literature search and observing demo:

- Helps in writing new parallel programs.
- For functional parallelization.
- Loops need to be transformed to subroutine calls for parallel execution.
- No help in dependency analysis.
- Static specification for parallelism only.
- Versions work for message-passing machines will be available at the end of summer, 1991.

Platform: CRAY2

OS: UNIX

Language: Fortran

GUI: X-Windows

Cost: Public domain

Supplier: Univ. of Tennessee

Contact: Jack Dongarra, (615) 974-8295

Local contact: Doreen Cheng, (415) 604-4361

See reference [16] [17]

2.2 Force

Function:

- Fortran extension for parallel programming

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Good performance has been reported for structure analysis problems.
- Uses static partitioning, one level fork-join parallelism only.

Platform: Y-MP, CRAY2, Encore, Sequent, Convex, Alliant

OS: UNIX

Language: Fortran

GUI: None

Cost: Public domain

Supplier: Univ. of Colorado

Contact: Harry Jordan, (303) 492-1411

Local access: Doreen Y. Cheng (415) 604-4361

See reference [18] [19] [20]

2.3 ParaScope

Function:

- Dependency analysis
- Code transformation
- DO-loop parallelization

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Supplier plans to integrate performance visualization tools of SCHEDULE into ParaScope.
- Debugging facilities are in development. These tools may require the program to be written in PCF Fortran.
- Supplier plans to develop tools to allow a user to annotate a program written for shared-memory machines and automatically translate it to message-passing.

Platform: SUN

OS: UNIX

Language: Fortran

GUI: X-Windows

Supplier: Rice Univ.

Contact: Ken Kennedy, (713) 285-5186

Local contact: Doreen Cheng, (415) 604-4361

See reference [21] [22] [23]

2.4 PAT

Function:

- Program instrumentation
- Performance analysis
- Parallel debugger
- Static code analysis
- Interactive parallelization

Evaluation by Kathi Flecher and Doug Pase through use:

- Has serious bugs.
- Lacks many claimed functions.

Platform: CRAY

OS: UNIX

Language: Fortran

GUI: X-Windows

Supplier: Georgia Institute of Technology

Contact: Kevin Smith, smith@boa.gatech.edu

Local contact: Doreen Cheng, (415) 604-4361

See reference [24]

2.5 Axe/Aims

Function:

- Axe provides a *Parallel Program Behavior Description Language* for describing interactions between processes.
- Axe predicts the performance of the program.
- Aims instruments a Fortran program and collects the run time information.
- A visualization tool allows viewing the behavior and the run time information.

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- About to be released for use.

Platform: iPSC/860

OS: UNIX

Language: Fortran

GUI: X-Windows

Supplier: NASA Ames Research Center

Contact: Jerry Yan, (415) 604-4381

See reference [25]

2.6 PIE

Function:

- Language extensions for parallel programming
- Performance predictor
- Performance trace and visualization

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Non-standard language extensions
- Has potential to be more than a performance tuning tool.
- Difficult to port to UNIX

Platform: Sun, VAX, Encore

OS: Mach (distributed UNIX)

Language: C, Fortran, Ada

GUI: X-Windows

Supplier: Carnegie-Mellon Univ.

Contact: Zary Segall, (412) 268-3736

Local contact: Ann Patterson-Hine, (415) 604-4178

See reference [26] [27] [28]

2.7 Parti

Function:

- Provides a set of procedures to be called from C or Fortran program that will translate read/write into send/receive (transform shared-memory programs into message-passing).
- Schedules the processes onto iPSC/860.

Evaluation by Doreen Cheng through phone contacts with the supplier:

- Load balancing must be done by the user.
- Supplier plans to extend this by adding higher level tools.

Platform: iPSC/860

OS: UNIX

Language: C, Fortran

Cost: Public domain

Supplier: NASA Langley ICASE

Contact: Joel Saltz, (804) 864-2197

Local access: Doreen Y. Cheng (415) 604-4361

2.8 Paragraph/PICL

Function:

- PICL generates execution profile of a parallel program on message passing machines.
- Paragraph allows visualization of the data collected.

Evaluation by Doreen Cheng through e-mail with the supplier:

- Tools could be integrated into our environment.

Platform: Intel and Ncube hypercubes, Symult/Ametek, Cogent

Language: C, Fortran

GUI: X-Windows

Cost: Public domain

Supplier: Oak Ridge National Lab

Contact: Michael T. Heath, mth@indigo.EPM.ORNL.GOV

Local contact: Doreen Cheng, (415) 604-4361

2.9 Hypertask

Function:

- Provides comment-directives for writing C programs for Intel hypercube machines.
- Provides library calls for dynamically resizing arrays by a factor of 2 each call
- Automatically divides arrays and loop iterations among all nodes in a cube of any size at run time.
- Data locality is considered.
- Inserts message passing directives.
- Plots flops/processor in 3D (Sunview only).

Evaluation by Doreen Cheng through literature search and e-mail with supplier:

- The current version contains known bugs.

Platform: iPSC/860

OS: UNIX

Language: C

GUI: X-Windows, Sunview

Cost: Public domain

Supplier: Intel

Contact: Marc Baber, marc@isc.intel.com

Local contact: Doreen Cheng, (415) 604-4361

See reference [29]

2.10 CRAY/fpp

Function:

- Automatic DO-loop parallelization
- Code transformation to take advantage of CRAY architecture

Evaluation by Doug Pase and Katherine Fletcher through use:

- The code generated for NAS benchmarks on Y-MP is 90% parallelized or less.
- User interface is batch oriented.
- Uses static program analysis only.

Platform: CRAY machines

OS: UNICOS, COS

Language: Fortran

GUI: None

Supplier: Cray Research

See reference [30] [31]

3. Tools Worth Tracking

3.1 PPD

Function:

- Event trace and post-mortem analysis for debugging
- Race condition detection
- Data flow and dependency analysis for debugging

Evaluation by Doreen Cheng through literature search and e-mail with the supplier:

- Part of it will be available at the end of summer 1991.

Supplier: Univ. of Wisconsin

Contact: Barton P. Miller, (608) 263-3378

See reference [32] [33] [34]

3.2 Fortran-Linda

Function:

- Language extensions for parallel programming

Evaluation by Doreen Cheng through e-mail with the supplier:

- It is a proof-of-concept project not for distribution.

Platform: Encore

Contact: Nick Carriero, carriero-nicholas@CS.YALE.EDU

3.3 ParaScope/debugger

Function:

- Dependency analysis for debugging
- Static analysis of potential race conditions
- Instrumenting the code for race condition detection

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Not available yet

Supplier: Rice Univ.

Contact: Robert Hood, (713) 285-5182

3.4 Poker

Function:

- Graphic extension to C for parallel programming
- Trace and visualization of instruction execution through an emulator
- Visualization of data for debugging (Voyeur)

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Scaling is a problem since the process topology can only be defined statically.
- Poker project is replaced by a follow-up project ORCA which will incorporate the lessons learned.

Platform: iPSC (buggy), Sequent, Sun

OS: UNIX

Language: C

GUI: X11

Cost: $100

Supplier: Univ. of Washington

Contact: Larry Snyder, (206) 543-1695

See reference [35] [36]

3.5 CAPER

Function:

- Graphic language extension to C for parallel programming on message-passing machines
- A library of parallel algorithms
- A library of functions that convert distributed/parallel data structures into a target form
- Automatic generation of code to handle I/O
- Simple debugging facility

Evaluation by Doreen Cheng through phone contacts and e-mail with the supplier

- Not released yet
- Supplier will port to iPSC if required.

Platform: Sun, HPC (Bell Lab's machine)

OS: UNIX

Language: C

GUI: X-11

Cost: None (may change in the future)

Supplier: Bell Labs

Contact: Binay Sugla, (201) 949-0850

3.6 DINO

Function:

- C extension for data parallel programming
- Single program multiple data (SPMD)

Evaluation by Doreen Cheng through literature search and e-mail with the supplier:

- How well it works for CFD applications is not yet clear.
- No plan so far for Fortran.

Platform: Intel Hypercubes

OS: UNIX

Language: C

Cost: University distribution cost

Supplier: Univ. of Colorado

Contact: Bobby Schnabel, bobby@lupine.Colorado.EDU

See reference [37] [38]

3.7 Kali-Fortran

Function:

- Parallelizing compiler
- Using directives to express parallelism on message-passing machines

Evaluation by Doreen Cheng through phone contacts with the supplier:

- Prototype not completed yet

Supplier: NASA Langley ICASE

Contact: Joal Saltz, (804) 864-2197

3.8 FLO

Function:

- A graphic parallel programming language for scientific applications

Evaluation by Doug Pase:

- Propietary design by Floating Point Systems
- Very attractive conceptually
- No compiler available

Supplier: None

Contact: Martin Waugh (503) 629-7651 (Designer of FLO)

3.9 OACIS Tools

Function:

- A graphic parallel programming language for scientific applications based on ELGDF.
- Generates code for C-Linda and Strand88.
- When target machine topology and the task graph of a program is entered as input, a schedule of the run-order of each task is produced.
- Analyzes a sequential program and saves the analysis in a database.

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Still in research prototyping stage.
- Using graphic user interface for Linda and Strand88 can be more intuitive for scientists than the languages' original user interface.
- Currently the tools are on Macintosh only.

Platform: Macintosh

Supplier: OASIS

Contact: Tony Capitano, capitano@sunny.oacis.org

See reference [39]

3.10 Code/Rope

Function:

- Hierarchical graphic language to specify dependencies and firing rules of program components written in C or Fortran
- Insertion of parallel directives

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Is a research project.
- Could be used to learn the pros and cons of graphic parallel programming languages.

Cost: $200

Supplier: Univ. of Texas, Austin

Contact: James C. Browne, (512)-471-9579

See reference [40] [41]

3.11 KAP/CRAY

Function:

- Automatic DO-loop parallelization
- Code transformation to take advantage of CRAY architecture

Evaluation by Doug Pase and Katherine Fletcher through use:

- The code generated for NAS benchmarks on Y-MP is 90% parallelized or less.
- User interface is batch oriented.
- Uses static program analysis only.

Platform: Y-MP, X-MP, Sun, Vax

OS: UNICOS, COS, UNIX, Ultrix

Language: Fortran

GUI: None

Cost: First copy $7,500/yr
Add'l copy $3,750/yr
Site license $15,000/yr

Supplier: Kuck and Associates

Contact: Davida Bluhm, (217) 356-2288

See reference[31]

4. Tools Eliminated

4.1 PISCES 2

Function:

- Extensions to Fortran for parallel programming
- Performance monitoring

Evaluation by Doreen Cheng through literature search and phone contacts with the supplier:

- Not on the platforms we have or anticipated having
- No plans for further development

Platform: FLEX/32 (shared-mem)

OS: UNIX

Language: Fortran

Supplier: University of Virginia

Contact: Terrence W. Pratt, (804) 982-2229

See reference [42] [43]

4.2 OLYMPUS

Function:

- Graphic extension to C for parallel programming

Evaluation by Doreen Cheng through literature search:

- Is an experimental system.
- Not on the platforms we have and anticipated having.
- Based on RPC.

Platform: Sun

OS: UNIX

Language: C

GUI: Sunview

Supplier: Univ. of Colorado

Contact: Garry J. Nutt

See reference [44]

4.3 Triplex:

Function:

- Program instrumentation and visualization

Evaluation by Doreen Cheng through literature search:

- Not on the platforms we have and anticipated having

Platform: Sun, NCUBE

OS: SunOS

Languages: Unknown

GUI: Sunview

Cost: $250

Supplier: Tufts Univ.

Contact: David Krumme (author of the reference)

See reference [45]

4.4 TOPSYS

Function:

- An object-oriented parallel programming tool
- Parallel debugging
- Performance monitoring
- Dynamic load balancing
- Process-processor mapping
- Program animation

Evaluation by Doreen Cheng through literature search:

- Unable to reach the supplier
- Out of US, difficult to collaborate

Platform: iPSC2

OS: Unknown

Language: C, Fortran

GUI: X11

Supplier: Univ. W. Germany

Contact: Thomas Bemmerl, bemmerl@lan.infomatik.tu-muenchen.dbp.de

See reference [46]

4.5 IC*

Function:

- Specification language for parallel/distributed systems

Evaluation by Doreen Cheng through literature search:

- The language is difficult to use.
- The system is very slow (requires special purpose hardware).
- Designed for communication protocols.

Platform: Unknown

OS: UNIX

GUI: Unknown

Supplier: Bell Communication Research, Morristown, NJ

Contact: E. Jane Cameron (author of the reference)

See reference [47]

4.6 VMMP:

Function:

- Language extension for programming on both shared-memory and message-passing machines
- Dynamic load balancing

Evaluation by Doreen Cheng through literature search:

- Has a routine call user interface.
- High efficiency claimed.
- Not able to contact the supplier.
- Out of US, difficult to collaborate.

Platform: Sun, MMX, IBM ACE

OS: UNIX, MACH

Language: C

Supplier: Tel-Aviv Univ., Israel

Contact: Eran Gabber (author of the reference)

See reference [48]

4.7 Simple/Care

Function:

- Discrete event simulator
- Code instrumentation and visualization

Evaluation by Doreen Cheng through literature search:

- For object-oriented languages
- For hardware system design

Platform: Unknown

OS: Unknown

Language: Common Lisp

Cost: Public domain

Supplier: Stanford Univ.

Contact: Nakul Saraiya (author of the reference)

See reference [49]

Acknowledgements

I would like to thank Doug Pase, Hal Barraclough, Horst Simon, and Russell Carter for helping with preparing this report.

References

1. "The Forge User's Guide Version 7.01," *Pacific-Sierra Research*, Dec. 1990.
2. Pacific-Sierra Research, *MIMDizer User's Guide Version 7.01.*
3. Doreen Y. Cheng and Douglas M. Pase, "Evaluation of Automatic and Interactive Parallel Programming Tools," *Submitted to Supercomputing'91 Conference.*
4. Parasoft, *Express Information Package.*
5. James C. Browne, *private communication.*
6. Numerical Algorithms Group, Inc., *VecPar_77 Tutorial Introduction and Reference Manual*, Dec. 1, 1989.
7. David Klappholz and Apostolos D. Kallis, "FATCAT: A Tool to Aid in Maintaining, Modifying, and Parallelizing Fortran Codes," *System/Software News, National Energy Research Supercomputer Center*, vol. 14 No. 8, Aug. 1990.
8. Vincent A. Guarna, Jr., Dennis Gannon, David Jablonowski, and Allen D. Malony, "Faust: An Integrated Environment for Parallel Programming," *IEEE Software*, pp. 20-27, July 1989.
9. Barton P. Miller, Morgan Clark, Jeff Hollingsworth, Steven Kierstead, Sek-See Lim, and Timothy Torzewski, "IPS-2: The Second Generation of a Parallel Program Measurement System," *IEEE Trans. on Parallel and Distributed Systems*, vol. 1 No. 2, April 1990.
10. Ian Foster and Stephen Taylor, in *Strand New Concepts in Parallel Programming*, Prentice Hall, 1989.
11. Ian Foster and Stephen Taylor, "Strand: A Practical Parallel Programming Tool," *MCS-P80-0889, Argonne National Laboratory.*
12. David Gelernter and James Philbin, "Spending Your Free Time," *Byte*, May 1990.
13. Sudhir Ahuja, Nicholas Carriero, and David Gelernter, "Linda and Friends," *IEEE Computer*, pp. 26-34, Aug. 1986.
14. Edward N May, "Portable Parallel Programming in a Fortran Environment," *Computer Physics Communications*, pp. 278-284, 1989.
15. Min-You Wu and Daniel D. Gajski, "Hypertool: A Programming Aid for Message-Passing Systems," *IEEE Trans. on Parallel and Distributed Systems*, vol. 1 No. 3, pp. 330-343, July 1990.
16. J. J. Dongarra and D. C. Sorensen, "SCHEDULE: Tools for Developing and Analyzing Parallel Fortran Program," *Tech. Memo 86, Argonne National Laboratory*, Nov. 1986.

17. J. J. Dongarra and D. C. Sorensen, "SCHEDULE Users Guide," *Argonne National Laboratory*, June 1987.
18. Harry Jordan, "The Force," *ECE Tech. Report 87-1-1*, Jan. 1987.
19. Harry F. Jordan, Muhammad S. Benten, Norbert S. Arenstorf, and Aruna V. Ramanan, *Force User's Manual*, March 1989.
20. Olaf O. Storaasli, D. T. Nguyen, and T. K. Agarwal, "Parallel-Vector Solution of Large-Scale Structural Analysis Problems on Supercomputers," *AIAA Journal*, vol. 28 No. 7, July 1990.
21. C. David Callahan, Keith D. Cooper, Robert T. Hood, Ken Knennedy, and Linda Torczon, "Parascope: A Parallel Programming Environment," *International Journal of Supercomputer Applications*, vol. 2 No. 4, pp. 84-99, Winter, 1988.
22. Alan Carle, Keith D. Cooper, Robert T. Hood, Ken Kennedy, Linda Torczon, and Scott K. Warren, "A Practical Environment for Scientific Programming," *IEEE Computer*, Nov. 1987.
23. Robert Hood, Ken Kennedy, and John Mellor-Crummey, "Parallel Program Debugging with On-the-fly Anomaly Detection," *Proc. of Supercomputing '90*, Nov. 1990.
24. Bill Appelbe and Kevin Smith, "Start/Pat: A Parallel-Programming Toolkit," *IEEE Software*, pp. 29-38, July 1989.
25. Jerry Yan, "Axe Environment for Concurrent System," *IEEE Software*, p. 25, May 1990.
26. Zary Segall and Larry Rudolph, "Pie (A Programming and Instrumentation Environment for Parallel Processing)," *IEEE Software*, pp. 22-37, Nov. 1985.
27. Ted Lehr, Zary Segall, S. F. Vrsalovic, E. Caplan, Alan L. Chung, and Charles E. Fineman, "Visualizing Performance Debugging," *IEEE Computer*, pp. 38-51, Oct. 1989.
28. Harren Harrison, "Tools for Multiple-CPU Environments," *IEEE Software*, p. 45, May 1990.
29. Marc Baber, "Hypertask: Automatic Array and Loop Partitioning on the iPSC," *Proceedings of the 24th Hawaii International Conference on System Sciences, Software Track, Issues in Parallel Programming*, 1991.
30. Cray Research, Inc., *CF77 Compiling System Volume 4: Parallel Processing Guide.*
31. Douglas M. Pase and Katherine E. Fletcher, "Automatic Parallelization: A Comparison of CRAY fpp and KAI KAP/CRAY," *NASA Ames NAS Technical Report, RND-90-010*, Nov. 1990.
32. Robert H. B. Netzer and Barton P. Miller, "On the Complexity of Event Ordering for Shared-Memory Parallel Program Executions," *Proc. of 1990 International Conf. on Parallel Processing*, pp. II-93 - II-97, 1990.

33. Robert H. B. Netzer and Barton P. Miller, "Detecting Data Races in Parallel Program Execution," *Proc. of the 3rd Workshop on Programming Languages and Compilers for Parallel Computing*, Aug. 1990.

34. Barton Miller and Jong-Deok Choi, "A Mechanism for Efficient Debugging for Parallel Programs," *Proc. of the SIGPLAN'88 Conf. on Programming Language Design and Implementation*, pp. 135-144, June 22-24, 1988.

35. Lawrance Snyder, "Parallel Programming and Poker Environment," *IEEE Computer*, pp. 27-36, July, 1984.

36. Gail A. Alverson, William G. Griswold, David Notkin, and Lawrence Snyder, "A Flexible Communication Abstraction for Nonshared Memory Parallel Computing," *Proc. of Supercomputing '90*, Nov., 1990.

37. Matthew Rosing, Robert B. Schnabel, and Robert P. Weaver, "The DINO Parallel Programming Language," *Tech. Report CU-CS-457-90, CS Dept. Univ. of Colorado at Boulder*, April 1990.

38. Thomas M. Derby, Elizabeth Eskow, Richard Neves, Matthew Rosing, Robert B. Schnabel, and Robert P. Weaver, "DINO 1.0 User's Manual," *Tech Report CU-CS-501-90, CS Dept. Univ. of Colorado at Boulder*, April 1990.

39. Robert Babb, II, Bella Bose, Warren Harrison, Ted Lewis, Virginia Lo, Sanjay Rajopadhye, Walter Rudd, and Shreekant Thakkar, "A Status Report: Parallel Programming Support Environment Research at Oregon State University," *Tech Report TR-PPSE-89-10.*

40. J. C. Browne, M. Azam, and S. Sobek, "CODE: A Unified Approach to Parallel Programming," *IEEE Software*, July 1989.

41. J. C. Browne, T. Lee, and J. Werth, "Experimental Evaluation of a Reusable-Oriented Parallel Programming Environment," *IEEE Trans. on Software Eng.*, vol. 16 No. 2, Feb. 1990.

42. Terrence W. Pratt, "The PISCES 2 Parallel Programming Environment," *NASA Contractor Report 178927*, July 1987.

43. Terrence W. Pratt, "PISCES 2 User's Manual," *NASA Contractor Report 178934*, July 1987.

44. Gary J Nutt, A. Beguelin, I. Demeure, S. Elliott, J. McWhirter, and B. Sanders, *OLYMPUS User's manual.*

45. David Krumme and Alva Couch, "Triplex Parallel-Execution Monitor," *IEEE Software*, May 1990.

46. Thomas Bermmerl, "An Integrated and Portable Tool Environment for Parallel Computers," *Proc. of IEEE International Conf. on Parallel Processing*, pp. 50-53, 1988.

47. E. J. Cameron, D. M. Cohen, B. Gopinath, W. M. Keese, L. Ness, P. Uppaluru, and J. R. Vollaro, "The IC* Model of Parallel Computation and Programming Environment," *IEEE Trans. on Software Engineering*, vol. SE-14 NO 3, pp. 317-326, March 1988.

48. Eran Gabber, "VMMP: A Practical Tool for the Development of Portable and Efficient Programs for Multiprocessors," *IEEE Trans. on Parallel and Distributed Systems*, vol. 1 No. 3, July 1990.

49. Nakul Saraiya, "Simple/Care Concurrent-Application Analyzer," *IEEE Software*, May 1990.

Functions of Parallel Programming Tools (Transform Sequential to Parallel)					
	Forge/MIMDiser	Express	E/SP	VecPar_77	FATCAT
Dependency Analysis	Y		Y	Y	Y
DO-Loop Parallelisation	Y	Y	Y	Y	
Functional Parallelisation	Y	Y	Y		
Vectorisation	Y			Y	
Directive Insertion	Y		Y	Y	
Automatic Code Transformation /Optimisation for Target Architecture					
User Directed Code Transformation	Y		Y	Y	
Query about Program Info	Y		Y		Y
Run Time Info Used for Guidance	Y	Y	Y		
Guidance in Parallelisation /Vectorisation					
Guidance in Code Transformation					
Guidance in Dependence Elimination					
Program Profiling	Y	Y	Y		
Performance estimate			Y		
Parallelism Estimate					
Use a Database	Y	Y	Y		
Interactive	Y	Y	Y	Y	
Window-based GUI	Y	Y	Y		

Functions of Parallel Programming Tools (Transform Sequential to Parallel Continued)					
	Faust	IPS-2	Strand88	C-Linda	MONMACS
Dependency Analysis	Y				
DO-Loop Parallelisation					
Functional Parallelisation					
Vectorisation					
Directive Insertion					
Automatic Code Transformation /Optimisation for Target Architecture					
User Directed Code Transformation					
Query about Program Info	Y				
Run Time Info Used for Guidance				Y*	
Guidance in Parallelisation /Vectorisation					
Guidance in Code Transformation					
Guidance in Dependence Elimination					
Program Profiling					
Performance estimate					
Parallelism Estimate	Y				
Use a Database					
Interactive	Y		Y	Y*	
Window-based GUI	Y		Y		

* Reduce message passing by relocate Linda run time routines
* For Shared-memory only

Functions of Parallel Programming Tools (Transform Sequential to Parallel Continued)					
	Hypertool	Schedule	Force	Parascope	PAT
Dependency Analysis				Y	Y
DO-Loop Parallelisation				Y	Y
Functional Parallelisation		Y			
Vectorisation					
Directive Insertion	Y				
Automatic Code Transformation /Optimisation for Target Architecture					
User Directed Code Transformation				Y	
Query about Program Info					
Run Time Info Used for Guidance		Y			
Guidance in Parallelisation /Vectorisation					
Guidance in Code Transformation					
Guidance in Dependence Elimination					
Program Profiling		Y			
Performance estimate					
Parallelism Estimate					
Use a Database					
Interactive	Y	Y		Y	Y
Window-based GUI		Y		Y	Y

Functions of Parallel Programming Tools (Transform Sequential to Parallel Continued)					
	Axe/Aims	PIE	Parti	Paragraph/PICL	Hypertask
Dependency Analysis					
DO-Loop Parallelisation					
Functional Parallelisation					
Vectorisation					
Directive Insertion					
Automatic Code Transformation /Optimisation for Target Architecture					
User Directed Code Transformation					
Query about Program Info					
Run Time Info Used for Guidance					
Guidance in Parallelisation /Vectorisation					
Guidance in Code Transformation					
Guidance in Dependence Elimination					
Program Profiling					
Performance estimate	Y				
Parallelism Estimate					
Use a Database					
Interactive	Y	Y		Y	
Window-based GUI	Y	Y		Y	Y

Functions of Parallel Programming Tools (Transform Sequential to Parallel Continued)				
	CRAY/fpp	KAP/CRAY	Poker	PPD
Dependency Analysis	Y	Y		Y
DO-Loop Parallelisation	Y	Y		Y
Functional Parallelisation				
Vectorisation	Y	Y		
Directive Insertion	Y	Y		
Automatic Code Transformation /Optimisation for Target Architecture	Y	Y		
User Directed Code Transformation	Y	Y		
Query about Program Info				
Run Time Info Used for Guidance				
Guidance in Parallelisation /Vectorisation				
Guidance in Code Transformation				
Guidance in Dependence Elimination				
Program Profiling				
Performance estimate				
Parallelism Estimate				
Use a Database				
Interactive			Y	
Window-based GUI			Y	

Functions of Parallel Programming Tools (Writing New Programs)					
	Forge/MIMDizer	Express	E/SP	VecPar_77	FATCAT
Language for Parallel Programming	Y		Y		
Automatic Data & Loop Distribution	Y	Y			
Task Scheduling		Y			
Dynamic Load Balancing		Y			
Code Frames	Y				
Consistency Checking	Y				
Support for Fortran 90		Y			
Dynamic Resising Arrays					
Hardware Configuration		Y			
Parallel I/O		Y			

Functions of Parallel Programming Tools (Writing New Programs Continued)					
	Faust	IPS-2	Strand88	C-Linda	MONMACS
Language for Parallel Programming			Y	Y	Y
Automatic Data & Loop Distribution					
Task Scheduling					
Dynamic Load Balancing					
Code Frames					
Consistency Checking				Y	
Support for Fortran 90					
Dynamic Resising Arrays					
Hardware Configuration					
Parallel I/O					

Functions of Parallel Programming Tools (Writing New Programs Continued)					
	Hypertool	Schedule	Force	Parascope	PAT
Language for Parallel Programming		Y	Y		
Automatic Data & Loop Distribution					
Task Scheduling	Y	Y			
Dynamic Load Balancing	Y	Y			
Code Frames					
Consistency Checking					Y
Support for Fortran 90					
Dynamic Resising Arrays					
Hardware Configuration					
Parallel I/O					

Functions of Parallel Programming Tools (Writing New Programs Continued)					
	Axe/Aims	PIE	Parti	Paragraph/PICL	Hypertask
Language for Parallel Programming		Y	Y		Y
Automatic Data & Loop Distribution					Y
Task Scheduling			Y		Y
Dynamic Load Balancing					
Code Frames					
Consistency Checking					
Support for Fortran 90					
Dynamic Resising Arrays				Y	
Hardware Configuration					
Parallel I/O					

Functions of Parallel Programming Tools (Writing New Programs Continued)				
	CRAY/fpp	KAP/CRAY	Poker	PPD
Language for Parallel Programming			Y	
Automatic Data & Loop Distribution				
Task Scheduling				
Dynamic Load Balancing				
Code Frames				
Consistency Checking				
Support for Fortran 90				
Dynamic Resising Arrays				
Hardware Configuration				
Parallel I/O				

Functions of Parallel Programming Tools (Performance Tuning & Debugging)					
	Forge/MIMDiser	Express	E/SP	VecPar_77	FATCAT
Performance Monitoring Visualisation	Y	Y			
Memory Access Visualisation		Y			
Critical Path Analysis and Visualisation					
Communication and Event Monitoring and Analysis	Y	Y			
Source Level Debugging		Y			
Race Condition Prediction					
Race Condition Detection					
Deadlock Prediction					
Deadlock Detection		Y			

Functions of Parallel Programming Tools (Performance Tuning & Debugging Continued)					
	Faust	IPS-2	Strand88	C-Linda	MONMACS
Performance Monitoring Visualisation	Y	Y	Y	Y	Y
Memory Access Visualisation				Y*	
Critical Path Analysis and Visualisation		Y			
Communication and Event Monitoring and Analysis				Y	
Source Level Debugging				Y	
Race Condition Prediction					
Race Condition Detection					
Deadlock Prediction					
Deadlock Detection					

* Tuple space usage

Functions of Parallel Programming Tools (Performance Tuning & Debugging Continued)					
	Hypertool	Schedule	Force	Parascope	PAT
Performance Monitoring Visualization		Y		Y	Y
Memory Access Visualization		Y			
Critical Path Analysis and Visualization		Y			
Communication and Event Monitoring and Analysis		Y			
Source Level Debugging					
Race Condition Prediction				Y	
Race Condition Detection				Y	
Deadlock Prediction					
Deadlock Detection					

Functions of Parallel Programming Tools (Performance Tuning & Debugging Continued)					
	Axe/Aims	PIE	Parti	Paragraph/PICL	Hypertask
Performance Monitoring Visualisation	Y	Y		Y	Y
Memory Access Visualisation					
Critical Path Analysis and Visualisation					
Communication and Event Monitoring and Analysis	Y			Y	
Source Level Debugging					
Race Condition Prediction					
Race Condition Detection					
Deadlock Prediction					
Deadlock Detection					

Functions of Parallel Programming Tools (Performance Tuning & Debugging Continued)				
	CRAY/fpp	KAP/CRAY	Poker	PPD
Performance Monitoring Visualisation			Y	
Memory Access Visualisation				
Critical Path Analysis and Visualisation				
Communication and Event Monitoring and Analysis				
Source Level Debugging				
Race Condition Prediction				
Race Condition Detection				Y
Deadlock Prediction				
Deadlock Detection				

Parallel Programming Tools (Status)				
	Forge/MIMDiser	Express	E/SP	VecPar_77
Platform	Y-MP X-MP CRAY2 iPSC Sun IRIS NEC*	iPSC NCUBE Sun PC Macintosh Y-MP**	Sun CRAY* IRIS***	Y-MP Sun
Operating system	UNIX UNICOS	UNIX DOS MacOS	UNIX	UNIX
Languages	Fortran	Fortran Fortran 90 C C++	Fortran	Fortran
GUI	X-Windows Sunview	X-Windows Sunview Postscript	X-Windows Sunview	None
Maturity	Bugs Fixed Quickly	Unknown	Unknown	Unknown
Cost	$28,050/wkst mnt: $4,500/yr $93,500/10 wkst $93,500/CRAY $93,500/iPSC Site License $158,950 mnt: 18,000/yr	$3,000/iPSC/860 $15,000/Y-MP $1,500/Network of Suns mnt :20% / yr	TBD	$2,500/wkst $19,500/Y-MP Site License $33,510/yr Perpetual $78,250
Supplier	PSR	ParaSoft	SES	NAG
Contact	John Levesque (916) 621-1600	Adam Kolawa (818) 792-9941	J. C. Browne (512) 474-4526	Sheila Caswell (708) 971-2337

wkst: workstation
mnt: maintenance
yr: year
TBD: to be determined

* in the process of negotiation
** beta test
*** port if desired

Parallel Programming Tools (Status Continued)				
	FATCAT	Faust	IPS-2	Strand88
Platform	CRAY2 Y-MP X-MP Sun	CRAY	Y-MP* Sequent	Y-MP iPSC Sequent Encore Sun
Operating system	UNIX	UNIX	UNIX	UNIX
Languages	Fortran	Fortran	Fortran C	Fortran C
GUI	None	X-Windows	X-Windows	X-Windows
Maturity	Unknown	Unknown	Unknown	Unknown
Cost	$50,000/source	$100/source	$300/source	$22,000/iPSC/860 mnt: $3,375/yr $3,000/Sun mnt: $900/yr Site License: $30,000-$40,000 30% GOV dscnt
Supplier	NEAT	U. of Illinois	U.of Wisconsin	Strand
Contact	David Klapphols (201) 420-5509	David Hammerslag (217) 244-0277	Barton Miller (608) 263-3378	Timothy Mattson (503) 690-9830

mnt: maintenance
yr: year
dscnt: discount

* Available in March 1991

Parallel Programming Tools (Status Continued)				
	C-Linda	MONMACS	Hypertool	Schedule
Platform	Y-MP* iPSC/860** Sequent Encore Sun IRIS IBM RS/6000 Appllo	iPSC/860	iPSC/2	CRAY2
Operating system	UNIX	UNIX	UNIX	UNIX
Languages	C Fortran***	Fortran**** C	C	Fortran
GUI	X-Windows	X-Windows	X-Windows	X-Windows
Maturity	Unknown	Unknown	Unknown	Unknown
Cost	$7,000/10wkst $20,000/iPSC2 Site License for ARC: $90,000	None	Univ. Distribution	None
Supplier	SCA	ANL	UC Irvine	U of Tenn.
Contact	Ellen Smith (203) 777-7442	Ewing Lusk (708) 972-7852	Daniel Gajski (714) 856-4155	Jack Dongarra (615) 972-8295

wkst: workstation
mnt: maintenance
yr: year
dscnt: discount
TBD: to be determined

* not fully debugged
** will be available soon
*** not directly supported, can be called from C
**** available in March 1991

Parallel Programming Tools (Status Continued)				
	Force	Parascope	PAT	Axe/Aims
Platform	Y-MP CRAY2 Encore Sequent Convex Alliant	Sun	CRAY	iPSC/860
Operating system	UNIX	UNIX	UNIX	UNIX
Languages	Fortran	Fortran	Fortran	Fortran
GUI		X-Windows	X-Windows	X-Windows
Maturity	Unknown	Unknown	Unknown	Unknown
Cost	None	Univ. Dist	Univ. Dist	None
Supplier	U. of Colo.	Rice U.	GIT	NASA ARC
Contact	Harry Jordan (303) 492-1411	Ken Kennedy (713) 285-5186	Kevin Smith smith@boa.gatech.edu	Jerry Yan (415)604-4381

Parallel Programming Tools (Status Continued)				
	PIE	Parti	Paragraph/PICL	Hypertask
Platform	Sun Encore VAX	iPSC/860	iPSC Ncube Ametek Cogent	iPSC/860
Operating system	Mach	UNIX	UNIX	UNIX
Languages	Fortran C Ada	Fortran C	Fortran C	C
GUI	X-Windows		X-Windows	X-Windows Sunview
Maturity	Unknown	Unknown	Unknown	Unknown
Cost	None	None	None	None
Supplier	CMU	NASA Langley	ORNL	Intel
Contact	Zary Segall (412) 268-3736	Joel Salts (804) 864-2197	Michael Heath mth@indigo.epm.ornl.gov	Marc Baber marc@isc,intel.com

Parallel Programming Tools (Status Continued)				
	CRAY/fpp	KAP/CRAY	Poker	PPD
Platform	CRAY	CRAY Sun Vax	Sun Sequent iPSC*	
Operating system	UNICOS COS	UNIX UNICOS COS ULTRIX	UNIX	UNIX
Languages	Fortran	Fortran	C	
GUI			X-Windows	
Maturity	Bugs Fixed	Bugs Fixed	Unknown	Unknown
Cost	CRAY Compiler	1st copy: $7,500/yr Add'l: $3,750/yr Site License: $15,000/yr	Univ. Dist	Univ. Dist.
Supplier	CRAY	K&A	U. of Washington	U. of Wisconsin
Contact		Davida Bluhm (217) 356-2288	Larry Snyder (206) 543-1695	Barton Miller (608) 263-3378

yr: year
Add'l: additional copy

* buggy

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

ARC 275 (Rev Feb 81)